一．线性分类函数

f(x,W) = W\*x + b

x为输入图像，W为多个参数，f(x,W)为输出是多个分类的概率。

比如:

x 图像的大小为 32 x 32 x 3，排成一列，尺寸为 3072 x 1；

共10个分类，输出f(x,W)为 图像判定为10个分类中每个类别的概率，尺寸为10 x 1；

因此，可以反推出多个参数的矩阵尺寸为 10 x 3072.

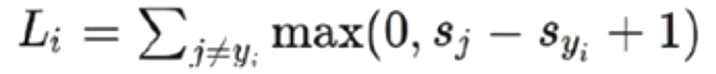
b是偏置项，尺寸为(10 x 1)

且通过矩阵的乘法公式可以得出，因为输出f(x,W)(10 x 1)为每个类别的概率，且每个类别的概率(1 x 1)是由参数矩阵的每一行向量(1 x 3072)，点乘x图像的列向量 (3072 x 1)而得来的。

参数矩阵的每一行向量(1 x 3072)就是图像的每个像素对应这个类别的3072个参数。一共有10行，即10个类别。可以说，图像的每个像素对应每个类别的参数都是不同的。

二. 损失函数

SVM损失函数



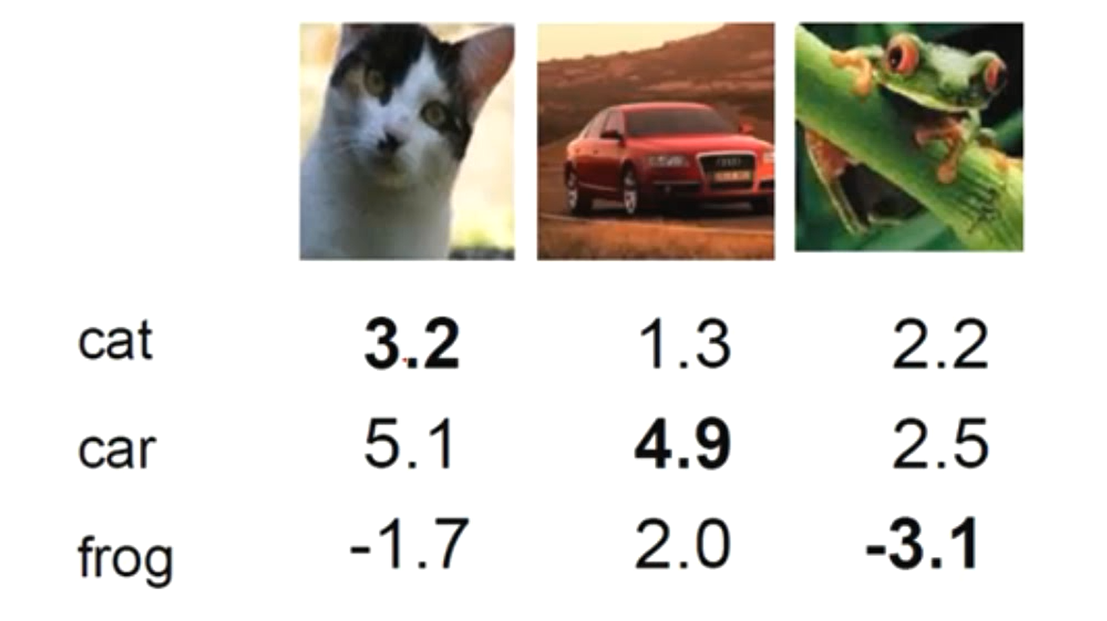
yi为正确分类，j为错误分类

计算的是图片属于正确分类的得分值与错误分类的得分值的差距。

后面加的1的意思是：只有错误分类的得分与正确分类的得分的相减值<-1时，才不会有损失值，这个值是自己设定的。一般设为 ∆，若错误分类分数小于正确分类，且差距大于∆，差距越大，越能说明分类效果好，越不需要损失值；若错误分类分数小于正确分类，但差距小于∆，说明分类效果不够好，还是需要设置损失值。

所谓的损失值为0的意思是，分类效果好。损失值越高，说明分类效果差。

举例：

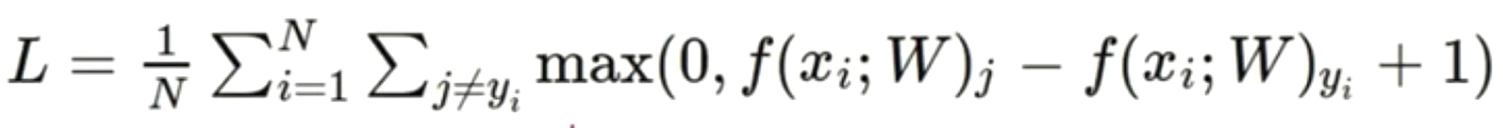


对于猫的分类的损失值：  
L1 = max(0, 5.1-3.2+1) + max(0, -1.7-3.2+1) = max(0, 2.9) + max(0, -3.9) = 2.9 + 0 = 2.9

对于车的分类的损失值：  
L2 = max(0, 1.3 – 4.9 +11) + max(0, 2.0-4.9\_1) = max(0, -2.6) + max(0, -1.9) = 0 + 0 = 0

对于青蛙的分类的损失值：  
L3 = max(0, 2.2 – (-3.1) +1) + max(0, 2.5 – (-3.1) +1) = max(0, 5.3) + max(0, 5.6) = 5.3 + 5.6 = 10.9

可以看出，分类效果最好的车损失值为0，分类效果最差的青蛙损失值为10.9



计算损失值不能用一张图片，而是要用N张图片，求取损失值的均值，排除样本个数对于结果的影响。

三. 正则化：

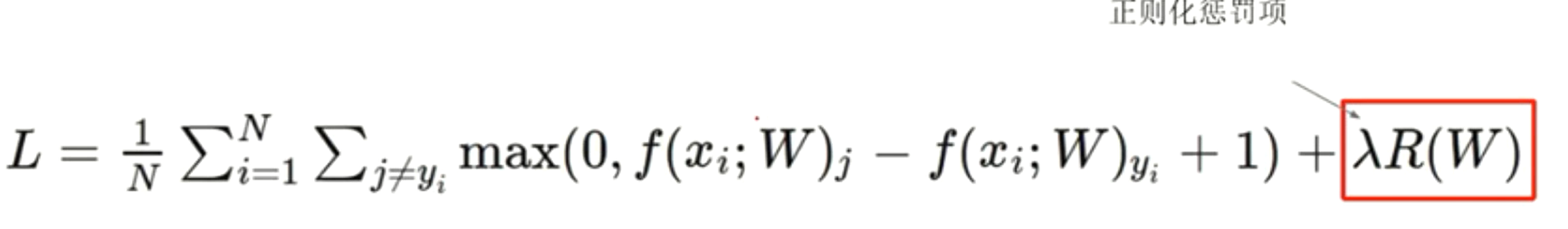
假设x = [1, 1, 1,1], W1 = [1, 0,0,0], W2 = [0.25.0.25.0.25.0.25]

可以计算得出 W1这个模型和W2这个模型乘以x得到的损失值是一样的。即不同的模型得出的损失值是一样的。

但是，可以明显看出，W1这个模型的权重只倾向于x的第一个像素，后面几个像素是什么与结果无关，很容易导致过拟合，这显然是不合理的。W2这个模型的权重比较均匀，分摊到各个像素点，比较合理。

那么，W1的损失值理应要比W2的损失值高，这样W1的模型才能有更大的改进。

因此，我们引入了正则化惩罚项：



假设惩罚项为L2惩罚项，则

对W1，1+ 0 + 0+0 = 1

对W2，1/16 + 1/16 + 1/16 + 1/16 = 1/4

则W1最终的损失函数会偏高一些，W2的损失函数会偏低一些。

下面是整合了正则化惩罚项的损失函数：

